JavaScript Developer's Toolchain

Nathaniel T. Schutta
@ntschutta

It all started so quietly...

No avoiding it now...

JavaScript is a first class citizen.

What does that mean?

How do we create modern web applications?

Applications are changing.

How do we embrace that?

It isn't the 90s anymore.

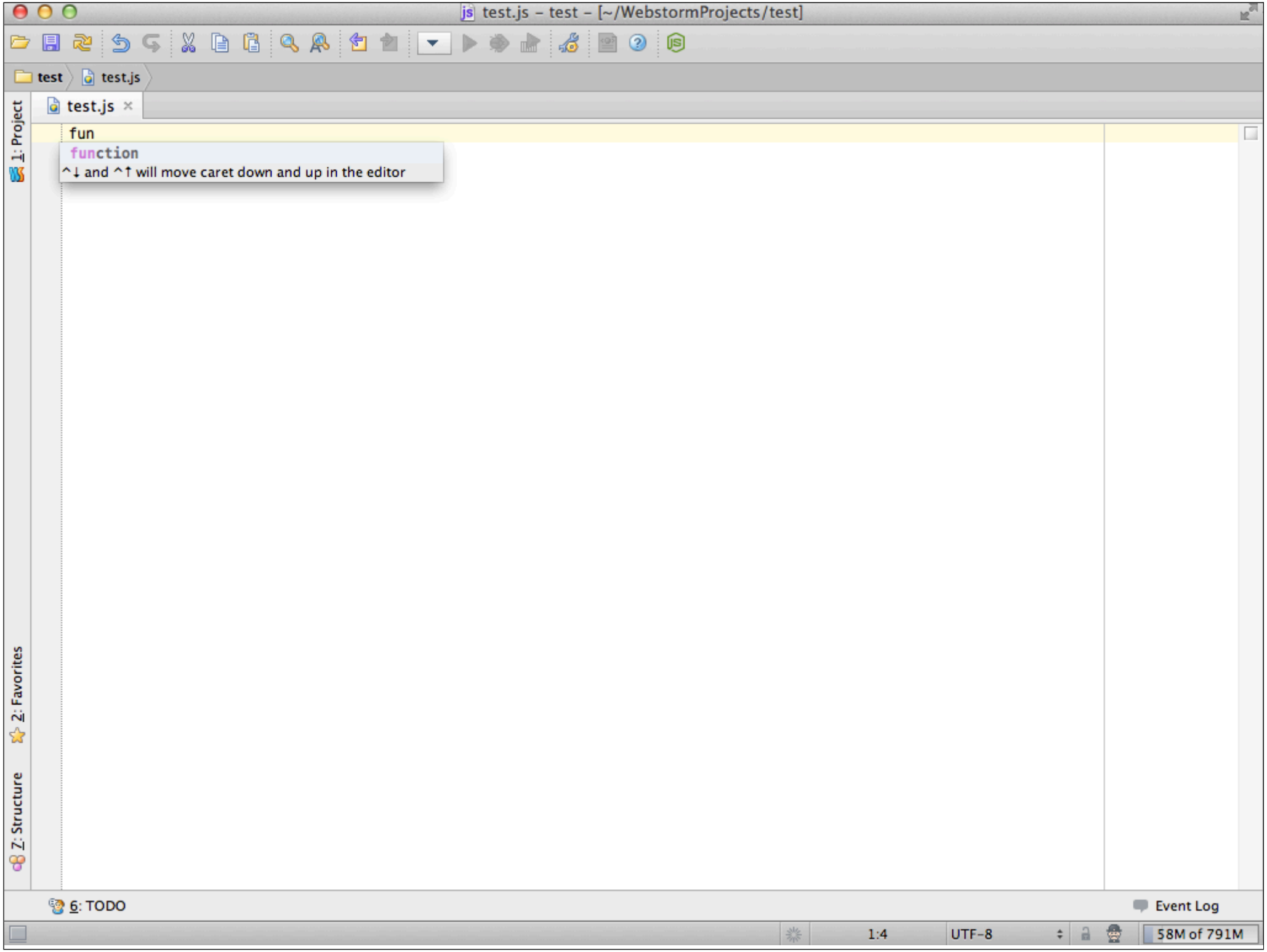
Remember alerts?

They aren't a best practice anymore.

One could argue they are considered harmful.

Today we have tools!

IDEs can do more than syntax highlighting...


test.js – test – [~/WebstormProjects/test]
test
test.js
test.js
fun
function
^↓ and ^↑ will move caret down and up in the editor
1: Project
2: Favorites
7: Structure
6: TODO
Event Log
1:4
UTF-8
58M of 791M

test.js – test – [~/WebstormProjects/test]

test > test.js

test.js

```
function foo () {
    v
}
```

var
void
DataView(bufferOrArrayOrLength, [N
DocumentView (DHTML.js)
ArrayBufferView() (HTML5.js)
ScriptEngineBuildVersion() String
ScriptEngineMajorVersion() String
ScriptEngineMinorVersion() String
SQLVoidCallback (HTML5.js)
HTMLVideoElement (HTML5.js)

^↓ and ^↑ will move caret down and up in the editor

1: Project
2: Favorites
7: Structure
6: TODO
Event Log
2:6
UTF-8
61M of 791M

But it doesn't end there.

Think of what you use in Java, C#, etc.

Testing, code coverage, build tools, code analysis...

Very rich toolset.

What about JavaScript?

JavaScript equivalents in nearly every case today.

In some cases,
paradox of choice.

We'll try and clear some of that up today.

This is a survey!

10,000 foot view.

We won't get into much detail...

Goal is to whet your appetite.

Show you what can be done today.

Give you some ideas.

Install them.

Get them running.

Cover the basics.

Here's what we *won't* do.

Deep dive on any of these tools.

Build a Node app.

How to test JavaScript.

Mocking/stubbing etc.

How to test ajax.

Evaluate JavaScript MV* frameworks.

Goal: setup the tools, gain some familiarity.

Node.js.

We need a
development platform.

Server side JavaScript.

Built on Google's V8 JavaScript engine.

Aka the one in Chrome.

It’s fast. Really fast.

Essentially a very lightweight app server.

Event driven,
non blocking IO.

Handles thousands of concurrent connections.

Made for real time apps.

Single threaded.

No locks.

Akin to Twisted or Event Machine.

Allows you to use JS on the client and server.

One language to unite them all?

Anyone using it that way?

More to it than just dealing with HTTP...

It's a platform.

Node Packaged Modules.

https://npmjs.org

AKA npm.

Makes it incredibly easy to publish and install things.

Even evil things!

The base of many tools in the JavaScript space today.

npm

https npmjs.org

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

- NODE.JS HOME
- DOWNLOAD
- ABOUT
- NPM REGISTRY
- DOCS
- BLOG
- COMMUNITY
- LOGOS
- JOBS

Search Packages

Create Account | Login

Node Packaged Modules

Total Packages: 44 345

5 802 094	downloads in the last day
31 421 029	downloads in the last week
108 106 611	downloads in the last month

Patches welcome!

Any package can be installed by using `npm install`.

Add your programs to this index by using `npm publish`.

Recently Updated

- 1m de
- 2m wintersmith-ls
- 2m metajs
- 3m scurvy
- 5m viceroy
- 6m blessed-life
- 6m xmpp-ftw-fanout
- 8m modella-resource
- 9m sockethub
- 10m svg-compactor

Most Depended Upon

- 4067 underscore
- 3107 async
- 2622 request
- 1936 optimist
- 1926 express
- 1821 commander
- 1674 coffee-script
- 1383 colors
- 1036 lodash
- 966 mkdirp

Incredibly useful.

Still has that new software smell.

Various platform dependent installs.

Lab time!

Install Node.js

- Install Node.js
- Go to http://nodejs.org click install or go to the download page
- Make sure that /usr/local/bin is in your $PATH
- Verify the install - at the command line:
  `node --version`

Node.js REPL

- At the command line:
  node
- At the prompt type:
  `a = [ ‘node’, ‘grunt’, ‘jasmine’]`
  `a.length`

“Joyent” Licensed.

node.js

nodejs.org

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Fork me on GitHub

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Current Version: v0.10.20

INSTALL

DOWNLOADS API DOCS

NODE.JS IN THE INDUSTRY

recruitics

Node allowed us to rapidly develop and deploy a low latency horizontally scalable solution in record time. We look forward to continuing to leverage node in key parts of

UBER

Node has allowed us to build a global, real-time logistics system without having to think twice about locking or concurrency issues.
Curtis Chambers

Transloadit

Node.js allows us to execute our many independent background processes in a non-blocking way. This is essential to make file uploading and encoding the

StrongLoop

Node.js is what our customers use to build enterprise-connected backends to mobile applications. The event-driven model and JavaScript for services orchestration makes

http://nodejs.org

Grunt.

We need a build tool.

JavaScript task runner.

Helps you automate.

Run tests, minify, linting...

Akin to make, Rake, Ant, Gradle, etc.

Large (and growing) plugin ecosystem.

Plugins for many everyday libraries/frameworks.

Create a package.json file and a Gruntfile.

Gruntfile can be JavaScript or CoffeeScript.

Package names
dependencies.

Gruntfile loads plugins and defines tasks.

Most everyday tasks exist as plugins.

Minification, linting, etc.

Can create custom tasks.

Still has that new software smell.

Installed via npm.

Lab time!

Install Grunt

- Install Grunt CLI
- From a command line (You may need to use sudo):
  `npm install -g grunt-cli`
- Verify the install - at the command line:
  `grunt --version`
- Change to the "app" directory
  ${extract}/js-dev-toolchain-workshop/app
- Execute grunt
  `grunt`

1. ntschutta@swan: ~/work/talks/js-dev/js-dev-toolchain-workshop/app (zsh)

```
app :: (master) » grunt     ~/work/talks/js-dev/js-dev-toolchain-workshop/app
grunt-cli: The grunt command line interface. (v0.1.11)

Fatal error: Unable to find local grunt.

If you're seeing this message, either a Gruntfile wasn't found or grunt
hasn't been installed locally to your project. For more information about
installing and configuring grunt, please see the Getting Started guide:

http://gruntjs.com/getting-started
app :: (master) »     ~/work/talks/js-dev/js-dev-toolchain-workshop/app 99 ↵
```

We need two things.

package.json - metadata

Gruntfile.js - define and configure tasks.

Both live in the root directory of your project.

Can create by hand.

But npm can help!

Lab time!

Setup Grunt

- From a command line:
  `npm init`
- Answer the questions!
- At the end you'll have a package.json file
- Add grunt - at the command line:
  `npm install grunt --save-dev`
- This will update your package.json file
- In the project root, create a new file called: Gruntfile.js (or Gruntfile.coffee)

Grunt File

- Add the wrapper function
```
module.exports = function(grunt) {
};
```
- Add project/task configuration
```
grunt.initConfig({
  pkg: grunt.file.readJSON('package.json')
});
```
- Load Grunt plugins (more on this later)
- Verify the configuration - at the command line:
```
grunt --force
```

1. ntschutta@swan: ~/work/talks/js-dev/js-dev-toolchain-workshop/app (zsh)

```
app :: (grunt-init*) » grunt --force
Warning: Task "default" not found. Used --force, continuing.

Done, but with warnings.
app :: (grunt-init*) »    ~/work/talks/js-dev/js-dev-toolchain-workshop/app
```

MIT License.

Grunt: The JavaScript Task Runner

gruntjs.com

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Getting Started Plugins Documentation

GRUNT

The JavaScript Task Runner

Latest Version

- Stable: v0.4.1
- Development: N/A

Ads by Bocoup.

Latest News

Grunt 0.4.1 released
March 13, 2013

Grunt 0.4.0 released

Why use a task runner?

In one word: automation. The less work you have to do when performing repetitive tasks like minification, compilation, unit testing, linting, etc, the easier your job becomes. After you've configured it, a task runner can do most of that mundane work for you—and your team—with basically zero effort.

Why use Grunt?

The Grunt ecosystem is huge and it's growi every day. With literally hundreds of plugins choose from, you can use Grunt to automat just about anything with a minimum of effort someone hasn't already built what you neec authoring and publishing your own Grunt pl to npm is a breeze.

Available Grunt plugins

http://gruntjs.com

Jasmine.

We must test our code.

BDD JavaScript
testing framework.

More readable?

Doesn’t require the DOM or other libraries.

Suites with specs.

Set of matchers.

Can create custom matchers.

Group related specs with '*describe*'.

Describe blocks can be nested.

Includes beforeEach and afterEach for setup/teardown.

Specs and blocks
can be disabled...

Don’t forget to
turn them back on!

Includes spies aka test doubles.

Can mock out the JavaScript clock.

Includes a *waitFor* to test asynchronous cases.

Runs in any JavaScript environment.

Include Jasmine file, page is now a runner.

Results read like a specification.

introduction-1.3.1.js

pivotal.github.io/jasmine/

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Passing 53 specs

No try/catch

```
A suite
  contains spec with an expectation

A suite is just a function
  and so is a spec

The 'toBe' matcher compares with ===
  and has a positive case
  and can have a negative case

Included matchers:
  The 'toBe' matcher compares with ===

  The 'toEqual' matcher
    works for simple literals and variables
    should work for objects
  The 'toMatch' matcher is for regular expressions
  The 'toBeDefined' matcher compares against `undefined`
  The `toBeUndefined` matcher compares against `undefined`
  The 'toBeNull' matcher compares against null
  The 'toBeTruthy' matcher is for boolean casting testing
  The 'toBeFalsy' matcher is for boolean casting testing
  The 'toContain' matcher is for finding an item in an Array
  The 'toBeLessThan' matcher is for mathematical comparisons
  The 'toBeGreaterThan' is for mathematical comparisons
  The 'toBeCloseTo' matcher is for precision math comparison
  The 'toThrow' matcher is for testing if a function throws an exception

A spec
  is just a function, so it can contain any code
  can have more than one expectation

A spec (with setup and tear-down)
  is just a function, so it can contain any code
  can have more than one expectation

A spec
  is just a function, so it can contain any code
  can have more than one expectation

  nested inside a second describe
    can reference both scopes as needed

A spy
  tracks that the spy was called
```

At least that's the idea ;)

Download the library.

Lab time!

Setup Jasmine

- Create a test folder in the application root
- In the test folder, create a lib and spec folder
- Expand the jasmine zip:
  ${extract}/js-dev-toolchain-workshop/installs/jasmine
- Copy the following to the lib folder created above:
  - jasmine.js
  - jasmine.css
  - jasmine-html.js
- In the spec folder, create a simple spec and spec runner

SpecRunner.html

file:///Users/ntschutta/work/talks/js-dev/js-dev-toolchain-workshop/app/test/spec/SpecRunner.

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Jasmine 1.3.0 revision 1354052693 finished in 0.018s

Passing 3 specs No try/catch

A suite
  contains spec with an expectation

A suite is just a function
  and so is a spec

Included matchers:
  The 'toBe' matcher compares with ===

MIT License.

introduction-1.3.1.js

pivotal.github.io/jasmine/

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Jasmine

Powered by PIVOTAL LABS

introduction-1.3.1.js

Jasmine is a behavior-driven development framework for testing JavaScript code. It does not depend on any other JavaScript frameworks. It does not require a DOM. And it has a clean, obvious syntax so that you can easily write tests.

This guide is running against Jasmine version 1.3.1 revision 1354556913.

Suites: `describe` Your Tests

A test suite begins with a call to the global Jasmine function `describe` with two parameters: a string and a function. The string is a name or title for a spec suite – usually what is under test. The function is a block of code that implements the suite.

Specs

Specs are defined by calling the global Jasmine function `it`, which, like `describe` takes a string and a function. The string is a title for this spec and the function is the spec, or test. A spec contains one or more expectations that test the state of the code under test.

An expectation in Jasmine is an assertion that can be either true

```
describe("A suite", function() {
  it("contains spec with an expectation", function(
    expect(true).toBe(true);
  });
});
```

http://pivotal.github.io/jasmine/

Mocha.

Another JS testing tool.

Perhaps “meta” tool.

Browser based.

Includes code coverage.

Highlights slow tests.

Detects globals.

Can run tests based on a regular expression...

Various reporting mechanisms.

Dot matrix, spec, TAP, landing strip, list, JSON...

Also includes JSON and HTML coverage reports.

Can use different assert libraries.

Allows for BDD or "assert" based testing.

Interface system supports various testing DSLs.

BDD, TDD, exports, QUint.

Supports asynchronous testing out of the box.

Can set test and
suite level timeouts.

Mocha looks for `/test/*.js`.

Works with sinon.js for spies, stubs and mocks.

Headless testing via PhantomJS.

Installed via npm.

MIT License.

Mocha – the fun, simple, flexible JavaScript test framework

visionmedia.github.io/mocha/

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

mocha

simple, flexible, fun

Mocha is a feature-rich JavaScript test framework running on **node.js** and the browser, making asynchronous testing simple and fun. Mocha tests run serially, allowing for flexible and accurate reporting, while mapping uncaught exceptions to the correct test cases. Hosted on **GitHub**.

Features

- browser support
- simple async support
- optionally run tests that match a regexp
- auto-exit to prevent "hanging" with an active loop

http://visionmedia.github.io/mocha/

Karma.

Tests need to be executed.

JavaScript test runner.

Created by AngularJS team.

Originated as MS thesis.

https://github.com/karma-runner/karma/raw/master/thesis.pdf

Quick feedback.

Works with several testing frameworks.

Works with continuous integration servers.

Change a file, Karma runs the tests.

Spawns a web server.

Results displayed at the command line.

Can also point a browser at the Karma web server.

Allows for mobile browser results.

Ships with Chrome and PhantomJS.

Plugins for all major browsers.

Can also point to custom browsers if needed.

Still has that new software smell.

Installed via npm.

Lab time!

Setup Karma

- Install Karma
- From a command line (you may need to use sudo)
  `npm install -g karma`
- Generate a Karma config file
- Change to the "app" directory
  ${extract}/js-dev-toolchain-workshop/app
- From a command line (you may need to use sudo)
  `karma init my.todo.js`
- Answer the questions

1. karma init my.todo.js (node)

```
Which testing framework do you want to use ?
Press tab to list possible options. Enter to move to the next question.
> jasmine

Do you want to use Require.js ?
This will add Require.js plugin.
Press tab to list possible options. Enter to move to the next question.
> no

Do you want to capture a browser automatically ?
Press tab to list possible options. Enter empty string to move to the next quest
ion.
> Chrome
>

What is the location of your source and test files ?
You can use glob patterns, eg. "js/*.js" or "test/**/*Spec.js".
Enter empty string to move to the next question.
> *.js
> test/**/*Spec.js
>

Should any of the files included by the previous patterns be excluded ?
You can use glob patterns, eg. "**/*.swp".
Enter empty string to move to the next question.
>

Do you want Karma to watch all the files and run the tests on change ?
Press tab to list possible options.
> yes
```

Launch Karma

- Launch Karma
- From a command line
  `karma start my.todo.js`
- Change a file, watch what happens
- Control-C to stop Karma

2. karma start my.todo.js (Google Chrome He)

```
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 1 of 3 SUCCESS (0 secs / 0.049 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 3 of 3 SUCCESS (2.932 secs / 0.05 s
ecs)
INFO [watcher]: Changed file "/Users/ntschutta/work/talks/js-dev/js-dev-toolchai
n-workshop/app/todos.js".
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 1 of 3 SUCCESS (0 secs / 0.016 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 2 of 3 SUCCESS (0 secs / 0.017 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 3 of 3 SUCCESS (0 secs / 0.018 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 3 of 3 SUCCESS (0.359 secs / 0.018
secs)
INFO [watcher]: Changed file "/Users/ntschutta/work/talks/js-dev/js-dev-toolchai
n-workshop/app/todos.js".
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 1 of 3 SUCCESS (0 secs / 0.017 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 2 of 3 SUCCESS (0 secs / 0.018 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 3 of 3 SUCCESS (0 secs / 0.018 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 3 of 3 SUCCESS (0.341 secs / 0.018
secs)
INFO [watcher]: Changed file "/Users/ntschutta/work/talks/js-dev/js-dev-toolchai
n-workshop/app/todos.js".
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 1 of 3 SUCCESS (0 secs / 0.016 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 2 of 3 SUCCESS (0 secs / 0.016 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 3 of 3 SUCCESS (0 secs / 0.016 secs
Chrome 31.0.1650 (Mac OS X 10.8.5): Executed 3 of 3 SUCCESS (0.309 secs / 0.016
secs)
```

MIT License.

Karma – Spectacular Test Runner for Javascript

karma-runner.github.io/0.10/index.html

Karma

Spectacular Test Runner for JavaScript

intro config plus dev about v0.10

> *On the AngularJS team, we rely on testing and we always seek better tools to make our life easier. That's why we created Karma - a test runner that fits all our needs.*

View project on GitHub

npm install -g karma

Testacular - JavaScript Test Runner

```
// list of files to exclude
exclude = [];

// use dots reporter, as travis terminal does not support escapir
// possible values: 'dots' || 'progress'
reporter = 'progress';
```

```
'use strict';

/* jasmine specs for controllers go here */
describe('PhoneCat controllers', function() {

  beforeEach(function(){
    this.addMatchers({
      toEqualData: function(expected) {
        return angular.equals(this.actual, expected);
      }
    });
  });


  beforeEach(module('phonecatServices'));



  describe('PhoneListCtrl', function(){
    var scope, ctrl, $httpBackend;

    beforeEach(inject(function(_$httpBackend_, $rootScope, $contr
      $httpBackend = _$httpBackend_;
      $httpBackend.expectGET('phones/phones.json').
          respond([{name: 'Nexus S'}, {name: 'Motorola DROID'}]);

      scope = $rootScope.$new();
      ctrl = $controller(PhoneListCtrl, {$scope: scope});
    }));
```

http://karma-runner.github.io/0.10/index.html

Phantom.

Browser testing can be slow and tedious.

Headless WebKit.

Headless testing in a CI environment.

Full DOM, CSS selector, JSON, Canvas and SVG.

Not itself a test runner...

Works with Jasmine, Mocha, QUnit, etc.

Several built in, most others have a 3rd party runner.

Can also capture screen shots.

Supports PNG,
JPEG, GIF and PDF.

Very useful for
diffing UI changes...

Can be used for network monitoring.

Page automation -
can even use jQuery.

Open a page, click a button...

Various platform dependent installs.

Lab time!

Setup PhantomJS

- Install Karma PhantomJS launcher
- From a command line (you may need to use sudo)
  `npm install karma-phantomjs-launcher --save-dev`
- Switch Karma to use Phantom instead of Chrome
  ${extract}/js-dev-toolchain-workshop/app/my.todo.js
- Launch Karma
- From a command line
  `karma start my.todo.js`
- Change a file, watch what happens
- Control-C to stop Karma

BSD License.

PhantomJS: Headless WebKit with JavaScript API

phantomjs.org

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Fork me on GitHub

PhantomJS

SOURCE CODE DOCUMENTATION API EXAMPLES FAQ

Full web stack
No browser required

PhantomJS is a headless WebKit scriptable with a JavaScript API. It has **fast** and **native** support for various web standards: DOM handling, CSS selector, JSON, Canvas, and SVG.

```
Simple Javascript example

console.log('Loading a web page');
var page = require('webpage').create();
var url = 'http://www.phantomjs.org/';
page.open(url, function (status) {
    //Page is loaded!
    phantom.exit();
});
```

Download v1.9

Get started

Community: Read the release notes Join the mailing list Report bugs

PhantomJS is an optimal solution for

HEADLESS WEBSITE TESTING
Run functional tests with frameworks such as Jasmine, QUnit, Mocha, Capybara,

SCREEN CAPTURE
Programmatically capture web contents, including SVG and Canvas. Create web site

PAGE AUTOMATION
Access and manipulate webpages with the standard DOM API, or with usual libraries like jQuery.

NETWORK MONITORING
Monitor page loading and export as standard HAR files. Automate performance analysis using YSlow

http://phantomjs.org

Sinon.

Simulating some test cases can be challenging.

Spies, mocks and stubs.

Spies observe.

Stubs replace a real object returning known things.

Mocks verify they're being used as expected.

No dependencies.

Runs anywhere JS runs.

Works with your favorite testing framework.

Allows you to test ajax interactions.

Can fake out XHR.

Also has a fake server.

Ability to fake time.

Adapters for Jasmine, QUnit etc.

Includes assertions and matchers.

Spies proxy calls, record what happened.

Can watch call order, frequency, arguments...

Exceptions, return values...

Stubs allow you to control the behavior.

Calls don't go to the underlying object.

Mocks set
expectations upfront.

Spy/Stub offspring
with expectations.

Several call modifiers.

Once, never, at least, at most, twice, etc.

Expectations about arguments, exceptions, etc.

What you expect from a stub/mock library.

With some JS specific bits.

Download the library.

BSD License.

Sinon.JS – Versatile standalone test spies, stubs and mocks for JavaScript

sinonjs.org

Sinon.JS

Standalone test spies, stubs and mocks for JavaScript.
No dependencies, works with any unit testing framework.

Documentation Help Issues Source code

Download Sinon.JS 1.7.3

2013-06-20 - Changelog - More builds/versions

6.0+ 2.0+ 7.0+ 10.10+ 3.2+ 3.2+ 2.1+ 0.2.x+

Getting started

The following function takes a function as it argument and returns a new function. You can call the resulting function as many times as you want, but the original function will only be called once:

```
function once(fn) {
    var returnValue, called = false;
    return function () {
        if (!called) {
            called = true;
            returnValue = fn.apply(this, arguments);
        }
        return returnValue;
    };
}
```

http://sinonjs.org

Istanbul.

We need to know
what we've tested...

And what we haven't.

Code coverage tool.

Bugs tend to lurk where there are no tests.

All JavaScript.

Tracks statement, branch and function.

Instrument on the fly.

Can run at the command line or as part of build.

HTML and LCOV reports.

Code coverage report for All files

gotwarlost.github.io/istanbul/public/coverage/lcov-report/index.html

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Code coverage report for **All files**

Statements: **96.14%** (1469 / 1528) Branches: **89.2%** (545 / 611) Functions: **97.96%** (336 / 343) Lines: **96.9%** (1436 / 1482)

	Statements		Branches		Functions		Lines	
istanbul/	100.00%	(3 / 3)	100.00%	(0 / 0)	100.00%	(0 / 0)	100.00%	(3 / 3)
istanbul/lib/	97.48%	(425 / 436)	89.76%	(184 / 205)	97.98%	(97 / 99)	98.12%	(417 / 425)
istanbul/lib/command/	94.63%	(194 / 205)	87.88%	(58 / 66)	89.58%	(43 / 48)	96.43%	(189 / 196)
/lib/command/common/	90.91%	(60 / 66)	84.85%	(28 / 33)	100.00%	(6 / 6)	92.31%	(60 / 65)
istanbul/lib/report/	94.78%	(436 / 460)	85.71%	(144 / 168)	100.00%	(94 / 94)	95.32%	(428 / 449)
istanbul/lib/store/	100.00%	(73 / 73)	85.71%	(12 / 14)	100.00%	(28 / 28)	100.00%	(72 / 72)
istanbul/lib/util/	97.54%	(278 / 285)	95.20%	(119 / 125)	100.00%	(68 / 68)	98.16%	(267 / 272)

Generated by istanbul at Tue Dec 04 2012 08:47:30 GMT-0800 (PST)

Still has that new software smell.

Installed via npm.

Lab time!

Setup Istanbul

- Install Istanbul
- From a command line (you may need to use sudo)
  `npm install -g istanbul`
- Change to the "app/test/spec" directory
  ${extract}/js-dev-toolchain-workshop/app/test/spec
- Execute the test
  `istanbul cover todoSpec.js`
- You should now have a coverage folder w/ files

Code coverage report for All files

file:///Users/ntschutta/work/talks/js-dev/js-dev-toolchain-workshop/app/test/spec/coverage/lcc

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Code coverage report for **All files**

Statements: **7.14%** (1 / 14) Branches: **100%** (0 / 0) Functions: **0%** (0 / 6) Lines: **7.14%** (1 / 14)

File	Statements		Branches		Functions		Lines	
spec/	7.14%	(1 / 14)	100.00%	(0 / 0)	0.00%	(0 / 6)	7.14%	(1 / 14)

Generated by istanbul at Wed Nov 13 2013 22:30:06 GMT-0600 (CST)

Run Jasmine via Grunt

- Install Grunt Jasmine runner
  `npm install grunt-contrib-jasmine --save-dev`
- Enable the plugin via your Gruntfile
  `grunt.loadNpmTasks('grunt-contrib-jasmine');`
- Add Jasmine template
- Execute Jasmine tests via Grunt
  `grunt jasmine`

2. ntschutta@swan: ~/work/talks/js-dev/js-dev-toolchain-workshop/app (zsh)

```
app :: (istanbul-init*) » grunt jasmine
Running "jasmine:todo" (jasmine) task
Testing jasmine specs via phantom
...
3 specs in 0.002s.
>> 0 failures

Done, without errors.
app :: (istanbul-init*) » ~/work/talks/js-dev/js-dev-toolchain-workshop/app
```

BSD License.

istanbul-js: A Javascript code coverage tool written in JS

gotwarlost.github.io/istanbul/

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Fork me on GitHub

istanbul: A Javascript code coverage tool written in JS

Yet another code coverage tool for Javascript, with the following features:

- All-javascript instrumentation library that tracks **statement, branch, and function coverage** and reverse-engineers **line coverage** with 100% fidelity.
- **Module loader hooks** to instrument code on the fly
- **Command line tools** to run node unit tests "with coverage turned on" and no cooperation whatsoever from the test runner
- **HTML** and **LCOV** reporting.
- Ability to use as **middleware** when serving JS files that need to be tested on the browser.
- Can be used on the **command line** as well as a **library**
- Based on the awesome *esprima* parser and the equally awesome *escodegen* code generator
- Well-tested on node 0.4.x, 0.6.x, 0.8.x and the browser (instrumentation library only)

Use cases

Supports the following use cases and more

- transparent coverage of nodejs unit tests
- ability to use in an `npm test` script for conditional coverage
- instrumentation of files in batch mode for browser tests (using yeti for example)
- Server side code coverage for nodejs by embedding it as custom middleware

Links

- Github repo
- Sample HTML reports: generated by the tool *for* the tool
- Sample LCOV reports: generated running the `genhtml` command from the `lcov` project on the lcov.info output
- Coverage object (JSON)
- API documentation

http://gotwarlost.github.io/istanbul/

JSHint.

Certain mistakes can be caught automatically.

Static code analysis.

Akin to JSLint.

http://www.jslint.com

Finds common mistakes and defects.

Very few programs pass ;)

Highly configurable.

Can configure inline, via file or in package.json.

Configuration is JSON.

Tune via ‘enforcing’ and ‘relaxing’ options.

Enforcing makes
JSHint more strict...

Relaxing does the opposite.

Dozens and dozens of configuration options.

http://www.jshint.com/docs/options/

Can also hide individual warnings if needed.

Installed via npm.

Lab time!

Setup JSHint

- Install JSHint
- From a command line (you may need to use sudo)
  `npm install jshint -g --save-dev`
- Change to the "app" directory
  ${extract}/js-dev-toolchain-workshop/app
- "hint" the JavaScript file
  `jshint todos.js`
- It should find some things it doesn't like ;)

2. ntschutta@swan: ~/work/talks/js-dev/js-dev-toolchain-workshop/app (zsh)

```
app :: (jshint-init*) » jshint todos.js
todos.js: line 67, col 19, Missing '()' invoking a constructor.
todos.js: line 232, col 17, Missing '()' invoking a constructor.

2 errors
app :: (jshint-init*) » ~/work/talks/js-dev/js-dev-toolchain-workshop/app 2 ↵
```

Run JSHint via Grunt

- Install Grunt JSHint plugin runner
  `npm install grunt-contrib-jshint --save-dev`
- Enable the plugin via your Gruntfile
  `grunt.loadNpmTasks('grunt-contrib-jshint');`
- Add JSHint task
- Execute JSHint via Grunt
  `grunt jshint`
- It should find some things it doesn't like ;)

2. ntschutta@swan: ~/work/talks/js-dev/js-dev-toolchain-workshop/app (zsh)

```
app :: (jshint-init*) » grunt jshint
Running "jshint:files" (jshint) task
Linting todos.js ...ERROR
[L67:C19] W058: Missing '()' invoking a constructor.
  var Todos = new TodoList;
Linting todos.js ...ERROR
[L232:C17] W058: Missing '()' invoking a constructor.
  var App = new AppView;

Warning: Task "jshint:files" failed. Use --force to continue.

Aborted due to warnings.
app :: (jshint-init*) » ~/work/talks/js-dev/js-dev-toolchain-workshop/app 3 ↵
```

Ratchet up the Heat!

- JSHint is highly configurable
- Enforce `===` usage
- Add `eqeqeq: true,` to the options array
- Execute JSHint via Grunt
  `grunt jshint`
- It should find more things it doesn't like ;)

2. ntschutta@swan: ~/work/talks/js-dev/js-dev-toolchain-workshop/app (zsh)

```
app :: (jshint-init*) » grunt jshint
Running "jshint:files" (jshint) task
Linting todos.js ...ERROR
[L67:C19] W058: Missing '()' invoking a constructor.
  var Todos = new TodoList;
Linting todos.js ...ERROR
[L130:C21] W116: Expected '===' and instead saw '==='.
      if (e.keyCode == 13) this.close();
Linting todos.js ...ERROR
[L211:C21] W116: Expected '!==' and instead saw '!='.
      if (e.keyCode != 13) return;
Linting todos.js ...ERROR
[L232:C17] W058: Missing '()' invoking a constructor.
  var App = new AppView;

Warning: Task "jshint:files" failed. Use --force to continue.

Aborted due to warnings.
app :: (jshint-init*) » ~/work/talks/js-dev/js-dev-toolchain-workshop/app 3 ↵
```

JSLint License...MIT with one addition:

“This Software shall be used for Good, not Evil.”

JSHint, a JavaScript Code Quality Tool

www.jshint.com

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

```
1   // Hello.
2   //
3   // This is JSHint, a tool that helps to detect errors and potential
4   // problems in your JavaScript code.
5   //
6   // To start, simply enter some JavaScript anywhere on this page. Your
7   // report will appear on the right side.
8   //
9   // Additionally, you can toggle specific options in the Configure
10  // menu.
11
12  function main() {
13    return 'Hello, World!';
14  }
15
16  main();
```

CONFIGURE

Metrics

There is only one function in this file.

It takes no arguments.

This function contains only one statement.

Cyclomatic complexity number for this function is 1.

JSHint

About

Documentation

Install

Hack

Blog

http://www.jshint.com

Plato.

Static analysis tool.

Source code visualizer, complexity analyzer.

Leverages JSHint and complexityReport.js.

Cyclomatic complexity,
line count, param count...

Halstead metrics,
maintainability index.

Plato – JavaScript Introspection

jsoverson.github.io/plato/examples/jquery/

Plato on Github | Report Home

Files

	complexity	sloc	est errors	lint errors
src/ajax/jsonp.js	13	80	0.56	0
src/ajax/script.js	7	56	0.31	0
src/ajax/xhr.js	15	111	0.71	0
src/ajax.js	134	855	6.34	2
src/attributes.js	109	505	4.57	7
src/callbacks.js	30	196	1.01	0
src/core.js	110	818	5.88	10

Includes detailed view.

jsoverson.github.io/plato/examples/jquery/files/src_core_js/index.html

jsoverson.github.io/plato/examples/jquery/files/src_core_js/index.html

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Plato on Github Report Home

function init

Complexity : 20
Length : 288
Difficulty : 36.34
Est # bugs : 0.56

```
    };

    // The ready event handler and self cleanup method
    completed = function() {
        veEventListener( "DOMContentLoaded", completed, false );
        EventListener( "load", completed, false );
        );

        ototype = {
        rsion of jQuery being used
        ion,

    constructor: jQuery,
    init: function( selector, context, rootjQuery ) {
        var match, elem;

        // HANDLE: $(""), $(null), $(undefined), $(false)
        if ( !selector ) {
            return this;
        }

        // Handle HTML strings
        if ( typeof selector === "string" ) {
            if ( selector.charAt(0) === "<" && selector.charAt( selector.length - 1 ) === ">" && selector.length
                // Assume that strings that start and end with <> are HTML and skip the regex check
                match = [ null, selector, null ];

            } else {
                match = rquickExpr.exec( selector );
            }

            // Match html or make sure no context is specified for #id
            if ( match && (match[1] || !context) ) {

                // HANDLE: $(html) -> $(array)
                if ( match[1] ) {
                    context = context instanceof jQuery ? context[0] : context;

                    // scripts is true for back-compat
                    jQuery.merge( this, jQuery.parseHTML(
                        match[1],
                        context && context.nodeType ? context.ownerDocument || context : document,
                        true
                    ) );

                    // HANDLE: $(html, props)
                    if ( rsingleTag.test( match[1] ) && jQuery.isPlainObject( context ) ) {
```

Still has that new software smell.

Installed via npm.

Lab time!

Setup Plato

- Install Plato
- From a command line (you may need to use sudo)
  `npm install plato -g`
- Change to the "app" directory
  ${extract}/js-dev-toolchain-workshop/app
- "plato" the JavaScript file
  `plato -d report todos.js`
- It should create a report

Plato – JavaScript Introspection

file:///Users/ntschutta/work/talks/js-dev/js-dev-toolchain-workshop/app/report/index.html

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

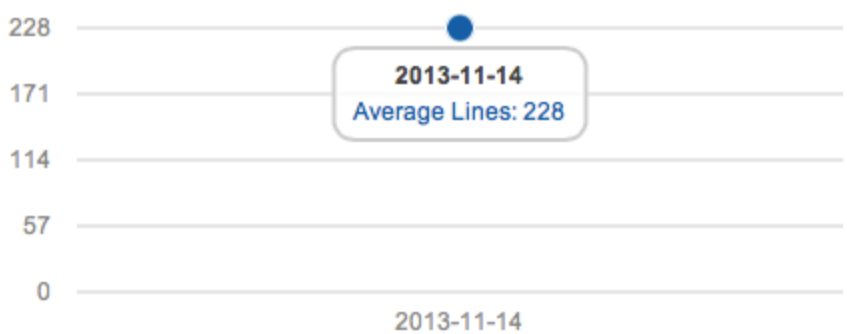
Plato on Github | Report Home

JavaScript Source Analysis

Summary

Total/Average Lines

228 / 228



Average Maintainability

70.72

Maintainability

Run Plato via Grunt

- Install Grunt Plato plugin
  `npm install grunt-plato --save-dev`
- Enable the plugin via your Gruntfile
  `grunt.loadNpmTasks('grunt-plato');`
- Add Plato task
- Execute Plato via Grunt
  `grunt jshint`
- It should create a new report

MIT License.

es-analysis/plato · GitHub

GitHub, Inc. github.com/es-analysis/plato

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

GitHub This repository Search or type a command

Explore Features Enterprise Blog

Sign up Sign in

es-analysis / plato

Star 1,196 Fork 49

JavaScript source code visualization, static analysis, and complexity tool

135 commits 4 branches 13 releases 7 contributors

branch: master plato /

Merge pull request #43 from ronkorving/patch-1 ...

jsoverson authored 24 days ago latest commit c994305c6e

bin	added lint reports on overview, new options, logger	9 months ago
lib	bootstrap 3 fixes	25 days ago
reports	report	25 days ago
test	+ add casperjs tests and update fixture 'b.js'	a month ago
.gitignore	initial support for collation and archival of report data	7 months ago
.jshintrc	jshint/complexity-report dependency update	5 months ago
.npmignore	added reports and .idea to npmignore	7 months ago
.travis.yml	+ update build process (travis.yml)	a month ago
Gruntfile.js	+ update $ grunt casper:test	a month ago
LICENSE-MIT	initial commit, testing d3 vs morris	10 months ago

Code

Issues 14

Pull Requests 0

Pulse

Graphs

Network

HTTPS clone URL

https://github.com/e

You can clone with HTTPS, or Subversion.

Clone in Desktop

Download ZIP

https://github.com/es-analysis/plato

Uglify.

Before deploying, we want to minimize the code.

JavaScript compressor.

Generates an AST of the JavaScript code.

Can manipulate the AST.

Can “beautify” compressed code for debugging.

Converts variable names to 1 character where possible.

Performs various optimizations.

Command line tool.

Also a JavaScript API.

Bit slower than YUI but smaller output.

Faster than Closure, output is about the same size.

Active development has shifted to UglifyJS2.

https://github.com/mishoo/UglifyJS2

Installed via npm.

Lab time!

Setup Uglify2

- Install Uglify2
- From a command line (you may need to use sudo)
  `npm install uglify-js -g`
- Change to the "app" directory
  ${extract}/js-dev-toolchain-workshop/app
- "uglify" the JavaScript file
  `uglifyjs todos.js --output dist/todos-min.js`
- It should create new minified file

Run Uglify via Grunt

- Install Grunt Uglify plugin
  `npm install grunt-contrib-uglify --save-dev`
- Enable the plugin via your Gruntfile
  `grunt.loadNpmTasks('grunt-contrib-uglify');`
- Add Uglify task
- Execute Uglify via Grunt
  `grunt uglify`
- It should create new minified file

BSD License.

mishoo/UglifyJS · GitHub

GitHub, Inc. github.com/mishoo/UglifyJS

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

GitHub This repository Search or type a command Explore Features Enterprise Blog Sign up Sign in

mishoo / UglifyJS

Star 4,361 Fork 404

JavaScript parser / mangler / compressor / beautifier library for NodeJS http://marijn.haverbeke.nl/uglifyjs

538 commits 2 branches 25 releases 36 contributors

branch: master UglifyJS /

v1.3.5

mishoo authored 5 months ago latest commit ef660f1bd0

bin	I'm officially switching to 4-space indentation.	a year ago
lib	Fix broken calls of complex binary expressions [#471]	a year ago
test	Fix broken calls of complex binary expressions [#471]	a year ago
tmp	Added an utility that allows walking depth-first easily.	2 years ago
.gitignore	add npm-debug.log to .gitignore	2 years ago
README.html	drop the pledgie link	11 months ago
README.org	drop the pledgie link	11 months ago
docstyle.css	NPM changes	3 years ago
package.json	v1.3.5	5 months ago
uglify-js.js	Consolidate null, Boolean, and String values	2 years ago

Code

Pull Requests 26

Pulse

Graphs

Network

HTTPS clone URL

https://github.com/m

You can clone with HTTPS, or Subversion.

Clone in Desktop

Download ZIP

https://github.com/mishoo/UglifyJS

Angular.

Managing large JavaScript applications hurts.

Developed by Google.

Used in production by Google today.

Trying to make up for lack of app support in HTML.

A shim until browsers reach their full potential.

Declarative binding.

Dependency injection.

Custom attributes: ng-.

Data binding updates view or model automatically.

Controllers abstract away the DOM.

Angular injects services into controllers.

Supports linking, bookmarking.

Includes form validation.

XHR wrapper: promises and exception handling.

Supports creating components via directives.

Supports localization.

As usual, a todo app...

```
<section id="todoapp" ng-controller="TodoCtrl">
  <header id="header">
    <h1>todos</h1>
    <form id="todo-form" ng-submit="addTodo()">
      <input id="new-todo" placeholder="What needs to be done?"
        ng-model="newTodo" autofocus>
    </form>
  </header>
```

```
todomvc.controller('TodoCtrl', function TodoCtrl($scope, $location, todoStorage, filterFilter) {
	var todos = $scope.todos = todoStorage.get();

	$scope.newTodo = '';
	$scope.editedTodo = null;

	$scope.$watch('todos', function () {
		$scope.remainingCount = filterFilter(todos, {completed: false}).length;
		$scope.completedCount = todos.length - $scope.remainingCount;
		$scope.allChecked = !$scope.remainingCount;
		todoStorage.put(todos);
	}, true);

	if ($location.path() === '') {
		$location.path('/');
	}

	$scope.addTodo = function () {
		if (!$scope.newTodo.length) {
			return;
		}

		todos.push({
			title: $scope.newTodo,
			completed: false
		});

		$scope.newTodo = '';
	};

	$scope.editTodo = function (todo) {
		$scope.editedTodo = todo;
```

Strong focus on testing.

Has its own mocks.

Includes a scenario runner.

Jasmine plugin.

Chrome plugin.

Doesn’t enforce a specific server architecture.

Doesn’t need to own the entire application.

Works with other libraries.

Good documentation.

Tutorials and examples.

Isn’t just used by Google.

Download the library.

MIT License.

AngularJS — Superheroic JavaScript MVW Framework

angularjs.org

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

ANGULARJS Home Learn Develop Discuss Search

HTML enhanced for web apps!

View on GitHub

Download (1.0.8/1.2.0-rc.3)

Follow +AngularJS on +1 +53315 Follow @angularjs 23.9K followers Tweet 4,210

Why AngularJS?

HTML is great for declaring static documents, but it falters when we try to use it for declaring dynamic

Alternatives

Other frameworks deal with HTML's shortcomings by either abstracting away HTML, CSS, and/or

Extensibility

AngularJS is a toolset for building the framework most suited to your application development. It is

http://angularjs.org

Bootstrap.

Many developers aren't skilled UI designers...

But we don't have budget for CSS experts.

Front end framework.

From Twitter.

CSS, JavaScript, themes...

Starts with attractive base that you can customize.

Highly customizable, build just the bits you want.

Includes glyphs and reusable components.

Dropdowns, navigation, progress bars, alerts, etc.

Mobile first.

Responsive.

CSS “starter kit”.

Flexible grid system.

Clean typography, tables, forms, buttons, images...

CSS designer in a box.

No CSS experts on staff?

Helps fill that gap.

Very popular, used by many sites and apps.

Uses Grunt.

Download kitchen sink or just what you need.

Apache 2.0 License.

Bootstrap

getbootstrap.com

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

Bootstrap Getting started CSS Components JavaScript Customize

Bootstrap

Sleek, intuitive, and powerful mobile first front-end framework for faster and easier web development.

Download Bootstrap

Star Fork Follow @twbootstrap 97.5K followe

Tweet 2,967

Currently v3.0.0 Bootstrap 2.3.2 docs GitHub project Examples Expo

http://getbootstrap.com

Require.

As JS apps grow, managing dependencies is critical.

File and module loader.

Manage dependencies.

Basically gives us import/ include ability.

Allows us to break our JS into discrete chunks.

In other words, modules.

Specifically Asynchronous Module Definition (AMD).

Allows asynchronous loading.

Shocking.

Define a code “unit”.

Define dependencies.

Hand crafting script tags doesn't scale well.

Devolves quickly...

Can this library be loaded before that one?

Grew out of Dojo and CommonJS.

Every approach
has drawbacks.

AMD has fewest.

Works well given browser limitations/restrictions.

AMD approach is used in a number of libraries today.

Require is compatible with all major browsers...

Even back to IE 6!

Also works in
Node or Rhino.

Can be used with jQuery and Dojo.

Download the library.

BSD or MIT License.

RequireJS

requirejs.org

Reader

html5 Gmail Firebug Lite Dropbox GitHub read later pinboard Instapaper

A JAVASCRIPT MODULE LOADER

Home

Start
Download
API
Optimization

Use with jQuery
Use with Node
Use with Dojo
CommonJS Notes

FAQs
Common Errors
Writing Plugins

Why Web Modules
Why AMD
Requirements
History

Get Help
Blog


```
/* ---

RequireJS is a JavaScript file and module
loader. It is optimized for in-browser use, but
it can be used in other JavaScript environments,
like Rhino and Node. Using a modular script
loader like RequireJS will improve the speed and
quality of your code.

IE 6+ .......... compatible ✔
Firefox 2+ ..... compatible ✔
Safari 3.2+ .... compatible ✔
Chrome 3+ ...... compatible ✔
Opera 10+ ...... compatible ✔

Get started then check out the API.

--- */
```

Latest Release: 2.1.9

Open source: new BSD or MIT licensed

web design by Andy Chung © 2011

http://requirejs.org

Is that all of them?

Nope.

Nearly every tool we looked at has alternatives.

Don’t like some of these?

Great, use what works for your team!

Perform a time boxed eval.

But use something!

Many of these tools are very new.

Expect some volatility.

Can’t treat JavaScript as a toy language today.

You don’t need to adopt every one of these today.

Pick your pain point.

Introduce one or two tools.

Ratchet up.

After a few weeks or months, add another.

Rinse repeat.

We've come a long way!

JavaScript is a first class citizen.

We have the
tools to prove it.

No excuses.

Thanks!

Nathaniel T. Schutta
@ntschutta